Generalized mixture models, semi-supervised learning, and unknown class inference

Samuel J. Frame · Sreenivasa Rao Jammalamadaka

Abstract In this paper, we discuss generalized mixture models and related semi-supervised learning methods, and show how they can be used to provide explicit methods for unknown class inference. After a brief description of standard mixture modeling and current model-based semi-supervised learning methods, we provide the generalization and discuss its computational implementation using three-stage expectation–maximization algorithm.

1 Introduction

Semi-supervised learning methods based on mixture models seek to improve classification results for known classes by using both labeled and unlabeled data (see for instance Dean et al. 2006). Although the resulting known-class inference is improved, it does not provide a solution for detecting unknown classes. At present, most methods for detecting unknown classes are based on ad hoc likelihood and goodness-of-fit comparisons to *tentative* known classes, and fail to provide explicit unknown class inference through either probability

statements or hypothesis tests. Semi-supervised learning methods are of great interest to the computer science, engineering, and statistics communities among others. For a good review and survey of literature see for instance Zhu (1996) and Miller and Uyar (1997).

The problem of detecting *unknown* classes is becoming increasingly more important in areas such as biology, medical diagnostics, and various defense applications. For instance, from a medical diagnostic perspective, the ability to dynamically detect new diseases or variations in pre-existing diseases is critical to reliable diagnosis, research, treatment, and patient recovery. Unknown class inference is critical to national defense as well as civilian law enforcement capabilities since adversaries are continually developing technologies to counter existing target recognition and tracking methods based on the known classes.

In most semi-supervised learning applications, one starts with a complete set of data X, which is a combination of *labeled* data and *unlabeled* data. Here, *labeled* data would mean that the class membership information, c_x is completely given as in the case of fully supervised learning. Class labels may be known through human inspection, experimental construction, or even inference from a learning algorithm. Let X_l represent the set of labeled data

$$X_\mathrm{l} = \{(x_1, c_{x_1}), \ldots, (x_{N_\mathrm{l}}, c_{x_{N_\mathrm{l}}})\}$$

where each *pair* consists of a vector observation, x_i, and the class label associated with the observation, c_{x_i}. Note, there is a total of N_l labeled data points. On the other hand *unlabeled* data may result from *new* data points (and/or data points randomly omitted for validation and testing purposes) which have yet to be analyzed or classified. As such, their class labels are unknown, hidden, or missing. Let X_u represent the set of unlabeled data

$$X_\mathrm{u} = \{(x_{N_\mathrm{l}+1}, m), \ldots, (x_N, m)\}$$

where each *pair* consists of a vector observation, x_i, and an indicator, m, that the observation is unlabeled or the class label is *missing*. Including these $(N - N_\mathrm{l})$ unlabeled data points, the complete set thus consists of the combined data $X = \{X_\mathrm{l}, X_\mathrm{u}\}$ for a total of N data points. Writing $f(x|\underline{\theta})$ for the density function (which is typically assumed to be either univariate or multivariate Gaussian in most cases), one can write the log-likelihood of the observed data for a standard mixture model with M components with weights α_k as

$$\log L_M(\Theta|X) = \sum_{x \in X_\mathrm{l}} \sum_{k=1}^{M} I(c_x = k) \log(\alpha_k f(x|\underline{\theta}_k)) + \sum_{x \in X_\mathrm{u}} \log \sum_{k=1}^{M} \alpha_k f(x|\underline{\theta}_k). \quad (1)$$

Here, the summation over the labeled data only contributes to the overall log-likelihood when the labeled data point originates from the appropriate generating component or class (see Dean et al. 2006; Miller and Uyar 1997). This is an extension of the standard mixture model and the parameter space consists of $\Theta = \{\underline{\alpha}, \{\theta_k\}_{k=1}^{M}\}$. In this scenario, the concept of a class is, typically, synonymous with a mixture component.

These models are lacking in some critical respects which the generalized mixture models (GMMs), which we describe in the next section, remedy. In the above description, the data are defined in such a way which only implicitly account for the fact that individual data points are either labeled or unlabeled. This information is contained in the class labels which are either c_x or m. However, we can explicitly define a random variable which indicates the presence or absence of a label. Let $L \in \{l, m\}$ where l and m denote the label is present or absent, respectively. In doing so, we have a larger set of data for which a generalization of Eq. (1) is needed. Although some of the previous semi-supervised methods provide an ad hoc discussion of unknown class inference, they do not investigate or explore new class discovery via probabilistic or statistical inference—what we would like to do more explicitly.

2 Generalized mixture models

Much like conventional semi-supervised learning methods, GMMs are constructed to handle a combination of both labeled and unlabeled data. *Generalizing* standard semi-supervised mixture models begins by incorporating a label presence or absence random variable, $L \in \{l, m\}$. As before, let $X = \{X_\mathrm{l}, X_\mathrm{u}\}$ be the combined labeled and unlabeled data. The N_l labeled data points now are a *triple* of the vector observation, the label indicator, and the class label so that

$$X_\mathrm{l} = \{(x_1, l, c_{x_1}), \ldots, (x_{N_\mathrm{l}}, l, c_{x_{N_\mathrm{l}}})\}$$

and the unlabeled set of data remains the same set of pairs

$$X_\mathrm{u} = \{(x_{N_\mathrm{l}+1}, m), \ldots, (x_N, m)\}$$

where m represents the fact that the class labels are missing or unlabeled.

Generalized mixture models explicitly seek to explain the additional label presence/absence information by way of model formulation. This information is a critical component which allows for explicit, probabilistic unknown class inference. To explain this information, GMMs allow for different types of mixture components. These components differ in how they generate labeled and/or unlabeled data points.

1. *Predefined components.* These components exclusively represent known classes. Predefined components generate data which are both labeled and unlabeled where we assume that the data labels, when missing, are *missing at random* (see Miller and Browning 2003; Little and Rubin 1987). The implication is that unlabeled data can originate from a known class. Note that more than one predefined component can represent a single known class.
2. *Non-predefined components.* These generate data which are exclusively unlabeled. As such, these mixture components will represent either the *outlier* regions of known classes or possibly unknown classes.

Let M be the number of mixture components in a GMM and let M_k denote the kth mixture component for $k = 1, \ldots, M$. Let C_{pre} denote the subset of components which are predefined components and the remaining subset of non-predefined components denoted by $\overline{C}_{\text{pre}}$. The mechanism by which GMMs explain the label presence/absence information is by probabilistically quantifying the rate at which a generic, predefined component will generate labeled data. Let M_{g} denote a generic, predefined component and $P(L = l|M_{\text{g}} \in C_{\text{pre}})$ be the rate or probability a predefined component generates labeled data. Note, this probability is the same for all predefined components such that this probability is "tied across all components" which are predefined as in Miller and Browning (2003). (Although one can extend this to allow for the label presence/absence probability to be specific to each class or component, clearly it is more messy and we will avoid doing that.) Since non-predefined components exclusively generate unlabeled data, $P(L = l|M_{\text{g}} \in \overline{C}_{\text{pre}}) = 0$.

For class representation, let P_c denote the set of all known classes where $c_x \in P_c$ for all of the labeled data. We allow for a probabilistic (or soft) ownership of known classes by predefined components. Predefined components are probabilistically associated with the set of known classes through a probability mass function (pmf),

$$\{P(C = c|M_k, L = l),\, c \in P_c\}_{M_k \in C_{\text{pre}}}.$$

For specific labeled data points, we have $P(C = c_x|M_k \in C_{\text{pre}}, L = l)$. Let α_k be the component weight which reflects the amount of data the components probabilistically owns. For now, let $f(x|\underline{\theta}_k)$ be a generic multivariate density function. Let

$$v_k = \begin{cases} 1 & \text{if } M_k \in C_{\text{pre}} \\ 0 & \text{if } M_k \in \overline{C}_{\text{pre}} \end{cases}$$

where the $\{v_k\}$ indicate each component as predefined or non-predefined. With this notation, the log-likelihood of the observed data for a model with M components is

$$\begin{aligned} \log L_M(\Lambda|X) = & \sum_{x \in X_{\text{l}}} \log \Bigg(\sum_{k=1}^{M} v_k \alpha_k f(x|\underline{\theta}_k) P(L = l|M_{\text{g}} \in C_{\text{pre}}) \\ & \qquad \times P(C = c_x|M_k \in C_{\text{pre}}, L = l) \Bigg) \\ & + \sum_{x \in X_{\text{u}}} \log \Bigg(\sum_{k=1}^{M} v_k \alpha_k f(x|\underline{\theta}_k) P(L = m|M_{\text{g}} \in C_{\text{pre}}) \\ & \qquad + (1 - v_k) \alpha_k f(x|\underline{\theta}_k) \Bigg). \end{aligned} \tag{2}$$

Given suitable amounts of data as well as the number of components, M, the parameters which must be learned in Eq. (2) are:

$$\begin{aligned}\Lambda &= \{\{\alpha_k\}_{k=1}^{M}, \{\underline{\theta}_k\}_{k=1}^{M}, P(L = l|M_g \in C_{\text{pre}}), \{P(C = c|M_k, L = l),\\ &\qquad c \in P_c\}_{M_k \in C_{\text{pre}}}, \{v_k\}_{k=1}^{M}\}\\ &= \{\Lambda_{\text{EM}}, \{v_k\}\}.\end{aligned}$$

Standard mixture models are usually discussed in the context of unsupervised learning using only unlabeled data, X_{u} (cf. Hastie et al. 2001; McLachlan and Krishnan 2004). In this situation, the only contribution to Eq. (2) comes from the summation over the unlabeled data. Although GMMs do not preclude the existence of known class model components, most methods do not hypothesize such model components a priori. As such, there are no predefined components and $v_k = 0$, for all $k = 1, \ldots, M$. The result is that Eq. (2) simplifies to

$$\log L_M(\Lambda|X_{\text{u}}) = \sum_{x \in X_{\text{u}}} \log \sum_{k=1}^{M} \alpha_k f(x|\underline{\theta}_k)$$

which is the standard mixture model used for model based clustering of unlabeled data points. With this example, one can see that GMMs are a generalized version of standard mixture models. Comparable simplifications occur when the data are restricted to only having a set of labeled data, X_{l}. We now turn our attention to estimating Λ in Eq. (2).

3 Semi-supervised learning

One existing method for estimating the model parameters is based on maximum likelihood. For fixed M, we use a generalized expectation–maximization (EM) algorithm (Miller and Browning 2003). The generalized EM algorithm consists of two steps: (a) choose the component natures, the $\{v_k\}$, to maximize Eq. (2) given all other parameters, Λ_{EM}, are held fixed and then (b) use EM to estimate Λ_{EM} given the $\{v_k\}$ are held fixed. As with EM, we are guaranteed to have non-decreasing $\log L_M(\Lambda|X)$. However, EM does not always guarantee convergence to global optima (McLachlan and Krishnan 2004).

3.1 Estimating the component natures

Depending on the size of the model as indicated by M, there are two ways to choose the component natures. If M is not too large, then one can enumerate all possible 2^M combinations of the component natures (each component nature either 0 or 1) and select the combination which maximizes $\log L_M(\Lambda|X)$ in Eq. (2). For large M, this strategy grows exponentially with M and is simply not feasible. A sub-optimal alternative (yet still having the property of a

non-decreasing L_M) is an iterative "one at a time" selection of the component natures (Miller and Browning 2003; Frame and Miller 2005). We cycle through the component natures and choose a single v_k to maximize $\log L_M(\Lambda|X)$ given all other component natures are held fixed. This is done for all the $\{v_k\}$ and this cyclic choosing is repeated until no more changes are made. Call this updated set of component natures $\{v_k\}^{(t+1)}$. The following EM method only uses the $\{v_k\}^{(t+1)}$ by defining the set of predefined and non-predefined components.

3.2 EM for the remaining model parameters

Recent technological advances have made expectation–maximization the standard maximum likelihood estimation (MLE) method for estimating the parameters of a mixture model (McLachlan and Krishnan 2004; Hastie et al. 2001). Expanding the EM framework for GMMs entails estimating and updating the label presence/absence probability, $P(L = l|M_{\mathrm{g}} \in C_{\mathrm{pre}})$, and the probability mass function over the known classes for the predefined components

$$P(C = c|M_k, L = l), \quad c \in P_c, \;\; M_k \in C_{\mathrm{pre}}.$$

The EM algorithm treats the observed data as incomplete. The needed missing information identifies which component generates each of the the labeled and unlabeled data points. Let V_{xk} be the *latent* indicator variable which indicates this information

$$V_{xk} = \begin{array}{ll} 1 & \text{if } x \in M_k \\ 0 & \text{if otherwise} \end{array}$$

where we require data to originate from a single component. With the $\{V_{xk}\}$ known, one can define a log-likelihood for the complete set of data

$$\begin{aligned} &\log L_C(\Lambda_{\mathrm{EM}}|X, \{v_k\}, \{V_{xk}\}) \\ &= \sum_{x \in X_{\mathrm{l}}} \sum_{k \in C_{\mathrm{pre}}} V_{xk} \log(\alpha_k f(x|\underline{\theta}_k) P(L{=}l|M_g \in C_{\mathrm{pre}}) P(C{=}c_x|M_k \in C_{\mathrm{pre}}, L{=}l)) \\ &\quad + \sum_{x \in X_{\mathrm{u}}} \sum_{k \in C_{\mathrm{pre}}} V_{xk} \log(\alpha_k f(x|\underline{\theta}_k) P(L = m|M_g \in C_{\mathrm{pre}})) \\ &\quad + \sum_{x \in X_{\mathrm{u}}} \sum_{k \in \overline{C}_{\mathrm{pre}}} V_{xk} \log(\alpha_k f(x|\underline{\theta}_k)). \end{aligned} \tag{3}$$

EM uses the complete log-likelihood in two distinct steps:

1. *Expectation* (E-step). Take the expected value of the complete log-likelihood given the current set of parameter estimates is held fixed. The expectation yields an expression with the expectation of the latent variable for

each sample. This step is the "ownership" step where we seek to find the probability of components owning a data point.

2. *Maximization* (M-step). Given the probabilistic association structure developed in the E-step, the M-step finds parameter estimates for the remaining parameters in the model which maximize the complete log-likelihood.

3.2.1 E-step

In the E-step, we have the expectation of the complete log-likelihood with respect to the latent variable, V_{xk} using $\Lambda^{(t)} = \{\Lambda_{\mathrm{EM}}^{(t)}, \{v_k\}^{(t+1)}\}$ which gives

$$E_{V_{xk}}[\log L_C(\Lambda_{\mathrm{EM}}|X, \{v_k\}, \{V_{xk}\})].$$

Since the V_{xk} are binary, one is left to solve

$$\begin{aligned} E[V_{xk}|x \in X_l, \Lambda^{(t)}] &= 1 \cdot P(V_{xk} = 1|x \in X_{\mathrm{l}}, \Lambda^{(t)}) + 0 \cdot P(V_{xk} = 0|x \in X_{\mathrm{l}}, \Lambda^{(t)}) \\ &= P(V_{xk} = 1|x \in X_{\mathrm{l}}, \Lambda^{(t)}) \end{aligned}$$

and $E[V_{xk}|x \in X_{\mathrm{u}}, \Lambda^{(t)}]$ follows similarly. Let $P(V_{xk} = 1|.) = P(x \in M_k|.)$ for notational convenience. It is easy to show with Bayes rule that these probabilities are given by

$$\begin{aligned} &P(x \in M_k|x \in X_{\mathrm{l}}, \Lambda^{(t)}) \\ &= \begin{cases} \dfrac{\alpha_k f(x|\underline{\theta}_k) P(C = c_x|M_k \in C_{\mathrm{pre}}, L = l)}{\sum_{k' \in C_{\mathrm{pre}}} \alpha_{k'} f(x|\underline{\theta}_{k'}) P(C = c_x|M_{k'}, L = l)} & \text{if } M_k \in C_{\mathrm{pre}} \\ 0 & \text{if } M_k \in \overline{C}_{\mathrm{pre}} \end{cases} \end{aligned} \tag{4}$$

and

$$\begin{aligned} &P(x \in M_k|x \in X_{\mathrm{u}}, \Lambda^{(t)}) \\ &= \begin{cases} \dfrac{\alpha_k f(x|\underline{\theta}_k) P(L = m|M_{\mathrm{g}} \in C_{\mathrm{pre}})}{\sum_{k' \in C_{\mathrm{pre}}} \alpha_{k'} f(x|\underline{\theta}_{k'}) P(L = m|M_{\mathrm{g}} \in C_{\mathrm{pre}}) + \sum_{k' \in \overline{C}_{\mathrm{pre}}} \alpha_{k'} f(x|\underline{\theta}_{k'})} & \text{if } M_k \in C_{\mathrm{pre}} \\ \dfrac{\alpha_k f(x|\underline{\theta}_k)}{\sum_{k' \in C_{\mathrm{pre}}} \alpha_{k'} f(x|\underline{\theta}_{k'}) + \sum_{k' \in \overline{C}_{\mathrm{pre}}} \alpha_{k'} f(x|\underline{\theta}_{k'})} & \text{if } M_k \in \overline{C}_{\mathrm{pre}} \end{cases} \end{aligned} \tag{5}$$

which are, essentially, the probability the M_k^{th} component generated a data point, x. Let $E[\log L_C]$ abbreviate the complete expression for the expected complete log-likelihood. If we substitute $E[V_{xk}|x \in X_{\mathrm{u}}, \Lambda^{(t)}]$ with $P(x \in M_k|.)$ we have

$$
\begin{aligned}
E[\log L_C] = & \sum_{x\in X_l}\sum_{k\in C_{\text{pre}}} P(x \in M_k | x \in X_l, \Lambda^{(t)}) \log(\alpha_k f(x|\underline{\theta}_k) \\
& \times P(L = l | M_g \in C_{\text{pre}}) P(C = c_x | M_k \in C_{\text{pre}}, L = l)) \\
& + \sum_{x\in X_u}\sum_{k\in C_{\text{pre}}} P(x \in M_k | x \in X_u, \Lambda^{(t)}) \log(\alpha_k f(x|\underline{\theta}_k) \\
& \times P(L = m | M_g \in C_{\text{pre}})) \\
& + \sum_{x\in X_u}\sum_{k\in \overline{C}_{\text{pre}}} P(x \in M_k | x \in X_u, \Lambda^{(t)}) \log(\alpha_k f(x|\underline{\theta}_k)). \qquad (6)
\end{aligned}
$$

3.2.2 M-step

To best demonstrate the M-step, we assume that $f(x|\underline{\theta}_k)$ is an r-dimensional multivariate normal distribution with mean vector and covariance matrix

$$
(\underline{\mu}_k, \Sigma_k) = \underline{\theta}_k
$$

say. In a model with M components, we are left to maximize $E[\log L_C]$ with respect to the remaining parameters

$$
\begin{aligned}
\Lambda_{\text{EM}}^{(t+1)} = \Big\{ & \{\alpha_k\}_{k=1}^M, \{\underline{\theta}_k\}_{k=1}^M, P(L = l | M_g \in C_{\text{pre}}), \\
& \{P(C = c | M_k, L = l), c \in P_c\}_{M_k \in C_{\text{pre}}} \Big\}
\end{aligned}
$$

using the probability structure generated in the E-step. For each component $k = 1, \ldots, M$, we have

$$
\alpha_k^{(t+1)} = \frac{\sum_{x\in X_l} P(x \in M_k | x \in X_l, \Lambda^{(t)}) + \sum_{x\in X_u} P(x \in M_k | x \in X_l, \Lambda^{(t)})}{N} \qquad (7)
$$

$$
\underline{\mu}_k^{(t+1)} = \frac{\sum_{x\in X_l} xP(x \in M_k | x \in X_l, \Lambda^{(t)}) + \sum_{x\in X_u} xP(x \in M_k | x \in X_l, \Lambda^{(t)})}{\sum_{x\in X_l} P(x \in M_k | x \in X_l, \Lambda^{(t)}) + \sum_{x\in X_u} P(x \in M_k | x \in X_l, \Lambda^{(t)})} \qquad (8)
$$

and the update for the covariance matrix follows similarly. These updates are natural extensions of the updates for the standard mixture model. The α_k reflect the amount of data owned by each component and μ_k are weighted averages as with standard mixtures. We must also update the label presence/absence and the class ownership probabilities for the predefined components. Updates of these probabilities are

$$
\begin{aligned}
& P(C = c | M_k \in C_{\text{pre}}, L = l) \\
& \quad = \frac{\sum_{x\in X_l : c_x = c} P(x \in M_k | x \in X_l, \Lambda^{(t)})}{\sum_{x\in X_l} P(x \in M_k | x \in X_l, \Lambda^{(t)})}, \quad c \in P_c, \;\; M_k \in C_{\text{pre}} \qquad (9)
\end{aligned}
$$

and

$$P(L = l|M_g \in C_{pre}) = \frac{\sum_{x \in X_l} \sum_{k \in C_{pre}} P(x \in M_k | x \in X_l, \Lambda^{(t)})}{\sum_{x \in X_l} \sum_{k \in C_{pre}} P(x \in M_k | x \in X_l, \Lambda^{(t)}) + \sum_{x \in X_u} \sum_{k \in C_{pre}} P(x \in M_k | x \in X_u, \Lambda^{(t)})} \quad (10)$$

which rely on the probability structure generated in the E-step. The class probabilities in Eq. (9) reflect the amount of labeled data from each class which are owned by each of the predefined components. The label presence/absence probability in Eq. (10) reflects the amount of labeled data owned by all predefined components.

The general outline of the semi-supervised learning method is as follows:

1. Learn the $\{v_k\}$ via cycling through them one at a time. Pick the individual value of v_k which maximizes $\log L_M(\Lambda|X)$ and repeat this process until no changes occur any more. Denote the updated set of component natures as $\{v_k\}^{(t+1)}$.
2. Use $\Lambda^{(t)} = \{\Lambda_{EM}^{(t)}, \{v_k\}^{(t+1)}\}$ and do EM learning until sufficient convergence has been achieved. Denote the updated set of parameters as

$$\Lambda^{(t+1)} = \{\Lambda_{EM}^{(t+1)}, \{v_k\}^{(t+1)}\}.$$

3.3 Model selection

Up until this point, we have developed GMMs based on the assumption that the number of components, M, is known. In this section we briefly describe how one can estimate the number of components in the mixture model. When fitting a standard mixture model with a learning method such as the generalized EM algorithm, the standard method by which M is selected is by BIC

$$\text{BIC} = \frac{1}{2} \log(N) \sum_{k=1}^{M} P_k - \log L_M(\Lambda|X)$$

where N is the number of data points, P_k is the number of parameters completely specifying component k, and $\log L_M(\Lambda|X)$ is the log-likelihood of a model specified by M components.

Computationally, choosing M this way is quite inefficient and time consuming. It requires that, for each value of M, models are extensively learned with the pre-described semi-supervised learning method. To reduce the computational burden, M should be bounded above by the number of components which are supported by the size of the data. Putting a lower bound on the number of components is to assume that each known class should be represented by at least

one component and consider at least one potential component for a possible unknown class.

Even with such restrictions, the number of possible models to explore can still be numerically overwhelming. One way to overcome this is to "overestimate" the number of components and then reduce the model size by a single component at a time (see e.g. Miller and Browning 2003). Such methods face problems such as how to best/optimally choose the component to eliminate (and thus, redistribute the ownership of associated data points and updating of model parameters). These methods typically do not produce reduced models which are "subsets" of the original larger model and rely on BIC evaluations to determine if the reduced model is better.

4 Inference and classification

We now consider the merit of GMMs for the purposes of inference. First, GMMs predict if an unlabeled sample belongs to a known or unknown class. Given a known class inference, GMMs can be used to predict which known class the unlabeled sample comes from (see Miller and Browning 2003).

The a posteriori probability that an unlabeled sample belongs to an unknown class is given by

$$P(M_{\mathrm{g}} \in \overline{C}_{\mathrm{pre}} | x \in X_{\mathrm{u}}) = 1 - \sum_{k \in C_{\mathrm{pre}}} P(x \in M_k | x \in X_{\mathrm{u}}, \Lambda^{(t)}) \tag{11}$$

where $P(x \in M_k | x \in X_{\mathrm{u}}, \Lambda^{(t)})$ is given in Eq. (5). Values of $P(M_{\mathrm{g}} \in \overline{C}_{\mathrm{pre}} | x \in X_{\mathrm{u}})$ greater than 0.5 suggest that the sample originates from an unknown class.

Given a *known class* inference for an unlabeled sample is made, the a posteriori known class probability is given by

$$\begin{aligned} &P(C = c | x \in X_{\mathrm{u}}, \Lambda^{(t)}) \\ &\quad = \frac{\sum_{k \in C_{\mathrm{pre}}} \alpha_k f(x | \underline{\theta}_k) P(C = c | M_k \in C_{\mathrm{pre}}, L = l)}{\sum_{k \in C_{\mathrm{pre}}} \alpha_k f(x | \underline{\theta}_k)}, \quad c \in P_c. \end{aligned} \tag{12}$$

Unlabeled samples are assigned to the class for which this probability is the largest.

5 Implementation

In this section, we discuss the computational implementation in relation to an original application. Since it is nearly impossible to evaluate the unknown class detection and inference capabilities in a real-time application, we use statistical cross-validation methods to achieve this. To assess unknown class detection capabilities, we use a *leave-one-out* method whereby sequentially removing an entire class temporarily. In removing each class, we are able to simulate an

unknown class, do unknown class discovery, and evaluate the ability of the learning method to find the unknown class. This process is repeated for all of the classes and the summary characteristics are compiled. Also, we provide standard errors (SE) of reported characteristics via *Monte Carlo* simulation where unlabeled (testing) samples are drawn at random from the available known classes. In reality, the situation is far different from the scenario we present here. In operational circumstances, all available labeled/known data is used in conjunction with *new* data we wish to learn about and classify. The results cannot be evaluated until human inspection and insight is used to determine if the automated classification process has succeeded.

To evaluate the machine learning process, we consider several summary characteristics. The first summary statistic is a classification matrix (often called a confusion matrix) which describes how the samples in each class are classified according to the possible classes available in the database. There are three summary statistics we use to evaluate the performance.

- $P(CC|Known)$. Given unlabeled samples are classified as belonging to a known class and are truly known, this is the probability that it was correctly classified (CC) to its known class. The error of classifying a known as an unknown is summarized in the next metric.
- $P(Unknown|Known)$. Given unlabeled samples belong to a known class, this is the probability they are misclassified as *unknown*.
- $P(Known|Unknown)$. This is the opposite error. Given unlabeled samples are unknown, this is the probability that they are misclassified as *known*. Precisely which known class is irrelevant. However, this information is available in the confusion matrix.

For each of these characteristics, we also provide an SE for the average of these statistics. The SE is estimated from the variability in the *Monte Carlo* simulations used. An example of such characteristics is found in Table 2 and an example of the classification matrix is found in Table 3.

Using BIC, the number of components for each class is chosen independently of one another. Then, all of the class-model components are merged into a single model. This final model is learned in the presence of all labeled data. This allows for the model components to change class associations and alter model parameters.

Semi-supervised learning extends this by learning *unknown* class model components based on the unlabeled data. Initially, this is done independently of the known class components. The components learned using unlabeled data are *tentatively* associated with the *unknown class*. Since the unlabeled data may contain data whose class membership is *not* unknown, we stress these components are *tentatively* associated with the *unknown class*. During the final learning process, all model components are merged together and semi-supervised learning is done with the complete set of unlabeled and labeled data. Unlabeled, known data points can become probabilistically associated with known class components via Eq. (5). This allows for unlabeled data points to be classified as *known* and to a particular known class. Also, these known class components

are allowed to change/update their parameters based on these unlabeled data points. This is how known class inference can be improved in semi-supervised learning. The remaining unlabeled data points (those which are not associated with known class components) stay associated with the *unknown* class. This allows for further *unknown* class refinement via updating of the non-predefined components based, hopefully, only on the unlabeled data points which are *unknown*.

6 Applications

As stated in the introduction our motivation for these methods came from from the perspective of (a) defense applications and (b) medical and biometric imaging applications. In this section, we show how the GMM's can be used in each of these scenarios to provide accurate identification or classification and reliable unknown class discovery and inference. We first discuss the example of classifying vehicles and then the application to bio-image informatics.

6.1 Vehicle recognition using hyperspectral data

This machine learning strategy and classification tool was used for classifying ground vehicles and unknown class inference in military applications (see e.g. Frame and Miller 2005). Since 2003, it has been used for Homeland Security applications and urban surveillance problems. In classical military *Automatic Target Recognition* (ATR) applications, many of the vehicle types are known and do not vary, even in large *Areas of Interest* (AOIs) for wide-area-automated surveillance applications. Now that paradigms are shifting towards being able to track and recognize civilian vehicles, even in small AOIs such as cities and small urban environments, being able to detect unknown vehicles is critical. For civilian vehicle recognition problems, there are substantially more vehicle manufacturers, makes, models, and variations than those for traditional military suppliers. The chance that particular vehicle types, models, and/or configurations are in an a priori known set of classes is highly unlikely. Being able to detect unknown or new vehicles is critical to the mission of Homeland Security and other domestic law enforcement agencies.

The vehicle recognition example that we consider uses *hyperspectral* data. In our example, there are three known vehicle types. The vehicles are a black Honda Civic, a red Pontiac Sunfire, and a silver Chevy Venture. Since the vehicles are of different colors, we would expect good classification results as well as good unknown class inference results. For these data, we run two experiments. The first experiment is a traditional, fully supervised learning paradigm. In this situation, full knowledge of the known classes is assumed and no unknown class discovery/inference is done. The second experiment is the desired unknown class discovery/inference using semi-supervised learning. In the semi-supervised case, we are actively looking for an unknown class. We evaluate the ability to detect unknown classes while not classifying *truly* known,

Table 1 Classification matrix for the supervised learning case, $P(CC|Known) = 0.9045$, Standarderror(SE) $= 0.0113$

Truth	Classified		
	Black Honda	Red Sunfire	Silver Chevy
Black Honda	0.7988	0.0018	0.1994
Red Sunfire	0.0022	0.9978	0
Silver Chevy	0.0643	0.0177	0.9180

unlabeled testing samples as unknown. For each class, there is a total of 130 or 120 samples. Since this does not present a rich data environment to utilize large mixture decompositions, we restrict the maximum number of components to represent each class to three components.

Table 1 contains the classification matrix for the supervised learning case. As we would expect, the data are classified quite well with the exception of the error which occurs in classifying the black Honda Civic as a silver Chevy Venture. The average rate of correct classification is about 90% with a small SE of about 1%. In these results, the only major mis-classification error is the aforementioned black Honda Civic classified as silver Chevy Venture. The reason for this is that although these colors are distinctly different to the human eye, a close inspection of the hyperspectral band data reveals that this data is rather similar. In fact, they seem to be almost indistinguishable as indicated in the graph in Fig. 1. The black lines are the feature vectors for the black Civic and the green lines are the features for the silver Chevy. It is clear that there is considerable overlap between the classes. With this consideration in mind, the error rate of 20% may actually seem rather small.

Tables 2 and 3 are the results for the semi-supervised case. From these results, it should be clear that unknown class discovery is working as we would like. When the data classes are *unknown*, our method discovers this almost 100% of the time. When known, unlabeled samples are classified into a known class,

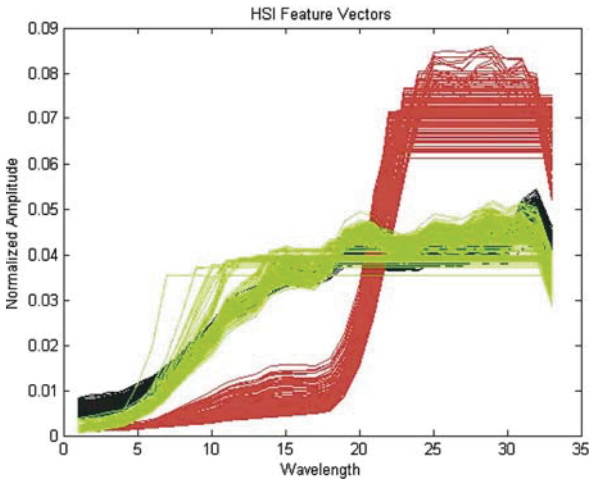


Fig. 1 Hyperspectral features: Black–Black Civic, Red–Red Sunfire, Green–Silver Chevy

Table 2 Summary metrics for the semi-supervised learning case

	P(CC|Known)	*P(Unknown|Known)*	*P(Known|Unknown)*
Average	0.9554	0.1471	0.0009
SE	0.002	0.09	0.055

Table 3 Classification matrix for the semi-supervised learning case

Truth	Classified			
	Black Honda	Red Sunfire	Silver Chevy	Unknown
Black Honda	0.6651	0	0.1214	0.2135
Red Sunfire	0.0002	0.9940	0.0003	0.0055
Silver Chevy	0.0067	0	0.7618	0.2315
Unknown	0.0008	0	0.0002	0.9991

they are correctly classified at a higher rate that when using standard fully supervised learning methods alone (0.09554 for semi-supervised versus 0.9054 for fully supervised). This increase is clearly not due to random deviation since the SE is in fact smaller (0.002 vs. 0.01). In fact, this is empirical evidence to suggest that the use of unlabeled samples in the learning process can improve known classification rates.

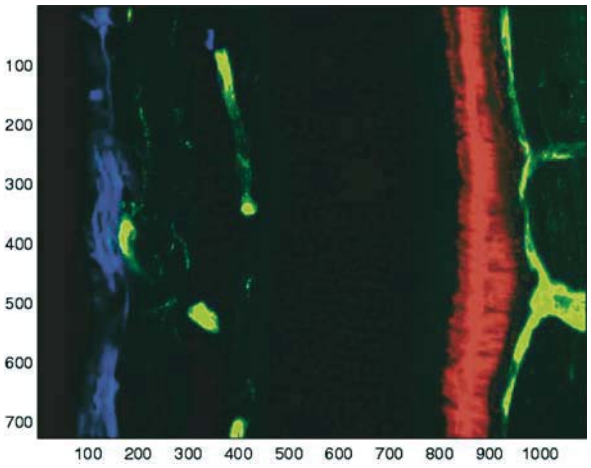
The drawback of the semi-supervised learning process, at least with these data, is that known, unlabeled data points are classified as *unknown* at a rate of approximately 0.1471 on average across all *leave-one-out* classes. For example, this misclassification rate is as high as 0.2315 for some vehicle classes. This is a result of the *leave-one-out* method when either the Honda Civic or the Chevy Venture are removed as an unknown class and the other class remains as a known class. In this situation, there is an unknown class which has a strong association with the respective known class allowing for the erroneous classifications. Still, this demonstrates that we can confidently identify unknown classes when they are present.

6.2 Bio-image informatics application

Researchers at the Neuroscience Research Institute (NRI) at the University of California, Santa Barbara, have various research programs that produce large numbers of biological digital images and related experimental data. The Information Technology Research (ITR) program to study Bio-Image Informatics is a collaborative effort funded by the National Science Foundation. For this analysis, we address one of the many components of this research effort.

One goal of the project is to study the effects of retinal detachment, reattachment, and any treatments that can be used. GMMs can be used to help understand differences between classes defined by normal retinas, length of retinal detachment, length of detachment, and the use of treatments. However,

Fig. 2 A normal retina image



here we only address the classification problem as a precursor to future medical diagnostic systems.

Retinal images are images of a cross-section or slice of a retina from a subject. The image is obtained with confocal microscopy or some other device. An example is found in Fig. 2. Generally, the subjects used in these experiments are small animals such as cats. Many feature extraction methods have been and are being developed to represent the retinal images. In the example we present here, we use a feature vector which represents the texture of the retina.

Gabor Filter analysis provides one method of developing feature vectors which capture the textures within images (see e.g. Manjunath et al. 2006). For instance in retinal images, we are looking for textures which capture the curvature, shapes, and contours of the cellular structures in different layers of the retina.

For this analysis, we focus on two classes defined by the normal retina (Normal) and 7 days of detachment (7 days). Since the set of comparable data is limited to only 29 and 27 samples for the respective classes, we must limit the maximum number of components to represent each class to 3. With such limited data, we expect to have high SE in correct classification rates. Table 4 is the classification matrix for this example.

For this experiment, the average correct classification rate is $P(CC|Known) = 0.7779$ with an SE of 0.1449. This is to be expected since the data is very limited and we do not have rich enough data to learn and represent each class. As more and more images come on stream, we expect this method to do better

Table 4 Confusion matrix: normal retina vs. 7 days of detachment

Truth	Classified	
	Normal	7 days
Normal	0.64	0.36
7 days	0.07	0.93

in classification and more importantly to identify unknown classes (like for instance an outlier, whose genesis can then be traced and investigated more carefully in the lab). For instance, a normal retina misclassified as one which is detached for 7 days poses a serious problem and allows doctors to conduct more tests and inspect the patient more thoroughly. This example highlights that GMMs are designed for large, rich data sources which contain representative feature vectors. As a medical diagnostic tool used for diagnosing problems with the retina, this example illustrates that the GMMs are capable of detecting undesirable medical conditions with good success.

7 Remarks

Representation of an image as a feature vector is clearly a very crucial and critical first step in both the examples we presented here. This is an area where an iterative interaction between the engineers who extract the features and the statisticians who use them for further analysis, is important.

The GMM presented in this paper assumes that components own classes in a probabilistic way. A subtle variation on this is to allow classes to own components in a deterministic way. This involves a slightly different parameter space, as well as a different EM method to obtain the parameters. A Bayesian/MCMC learning method needs to be developed and implemented for this alternative formulation, which is under investigation.

Acknowledgments The authors wish to thank Prof. David Miller of Penn State for helpful discussions and Prof. B. S. Manjunath of UCSB for providing the opportunity to collaborate on the Bio-Informatics project. The first author would also like to thank California Space Grant Graduate Fellowship Fund and Toyon Research Corporation for financial support and motivational problems.

References

Aitkin M, Rubin D (1985) Estimation and hypothesis testing in finite mixture models. J R Stat Soc Ser B (Methodol) 47:67–75

Dean N, Downey G, Murphy T (2006). Using unlabeled data to update classification rules with applications in food authenticity studies. J R Stat Soc Ser C (Appl Stat) 1–14

Frame S, Miller D (2005) Machine learning for robust automatic target recognition: phase 1—final report. Phase 1 Final Report for U.S. Air Force Research Laboratory Contract, FA8650-04-M-1659

Hastie T, Tibshirani R, Friedman J (2001) Elements of statistical learning. Springer, Heidelberg

Little R, Rubin D (1987) Statistical analysis with missing data. Wiley, New York

Manjunath BS, Newsam S, Wang L (2006) Using texture to analyze and manage large collections of remote sensed image and video. Appl Optics 43:2210–2217

McLachlan G, Krishnan T (2004) The EM algorithm and extensions. Wiley, New York

Miller D, Browning J (2003) A mixture model and EM-based algorithm for class discovery, robust classification, and outlier rejection in mixed labeled/unlabeled data sets. IEEE Trans Pattern Anal Mach Intell 1468–1483

Miller D, Uyar H (1997) A mixture of experts classifier with learning based on both labelled and unlabelled data. Adv NIPS 571–577

Schwarz G (1978) Estimating the dimension of a model. Ann Stat 6:461–464

Zhu X (1996) Semi-supervised learning literature survey. Tech. Report No. 1530, Computer Sciences Department, University of Wisconsin, Madison